COMP 2140 Assignment 5: BST, 2-3 Trees and Heap Sort

Helen Cameron

Due: Wednesday 7 December 2016 at 10:00 p.m.

How to Get Help

Your course instructor is helpful: Email me at Helen.Cameron@cs.umanitoba.ca or come to my office hours at 2:30–3:30 p.m. Mon/Wed/Fri or 4:00–4:30 p.m. Tues/Thurs in E2 477 EITC (or by appointment).

For email, please remember to put "[comp2140]" in the subject and use a meaningful subject, and to send from your UofM email account.

Course discussion groups on UM Learn: A discussion group for this assignment is available in the COMP 2140 course site on UM Learn (click on "Discussions" under "Communications"). Post questions and comments related to Assignment 5 there, and I will respond. Please do not post solutions, not even snippets of solutions there, or anywhere else. I strongly suggest that you read the assignment discussion group for helpful information.

Computer science help centre: The staff in the help centre can help you (but not give you assignment solutions!). See their website at http://www.cs.umanitoba.ca/undergraduate/computer-science-help-centre.php for location and hours. You can also email them at helpctr@cs.umanitoba.ca.

Programming Standards

When writing code for this course, follow the programming standards, available on this course's website on UMLearn. Failure to do so will result in the loss of marks.

General Overview of Assignment 5

Assignment 5 consists of three parts:

- Part A asks you to modify the solution to Assignment 4 to use a binary search tree, rather than a hash table (see details below in "Assignment Questions: Part A — Binary Search Trees").
- Part B asks you to answer some hand-written questions about 2-3 trees (see details below in "Assignment Questions: Part B — 2-3 Trees").
- Part C asks you to complete a nearly-complete application that uses heap sort to sort (see details below in "Assignment Questions: Part C — Heaps").

Since the assignment consists of three parts, the hand-in instructions have changed — make sure that you follow the hand-in instructions exactly.

Assignment Questions: Part A — Binary Search Trees

In this question, you must modify my solution to Assignment 4. My solution will become available on Friday November 25 after 11:00 p.m. (i.e., after the dropbox for Assignment 4 closes). It will be available on UM Learn in the same place you got the Assignment 4 question. You must change the solution to use a binary search tree implementation inside the `Table` class (instead of a hash table implementation) — that

is, a `Table` is a binary search tree, and the variable records are therefore stored in a binary search tree (still called a "`Table`", though).

Note that your solution must run on the same input files that you used for Assignment 4.

What must be changed in the `Table` class:

- You need to add a `BSTNode` class, which can either be a `private` class inside the `Table` class (with `public` instance variables) or a `public` class outside the `Table` class (with `private` instance variables). Note that you cannot use "`Node`" for the class name because that name is already in use as a linked-list node for the `Stack` class; therefore, use "`BSTNode`".
- The instance variables inside the `Table` class must be changed and the constructor updated to handle the new instance variable(s). Note that the constructor no longer needs any parameters.
- The bodies of the `public` methods must change to use a binary search tree, not a hash table. (But the headers of the `public` methods must not be changed). Note that method `printAll()` must print the variable records using an inorder traversal.
- The `private hash` method should be deleted.

Outside of the `Table` class and its associated `BSTNode` class, only the following may be changed:

- The name of the file must be changed to `A5PartA<yourName><yourStudentID>.java` and the name of the application class must be changed to match it.
- The call to the `Table` constructor in method `processAllAssignmentStatements()` must be changed to pass no arguments (because the constructor should no longer have any parameters).
- The header titles printed out in `main()` must be changed to titles appropriate to Assignment 5 Part A (i.e., not Assignment 4).

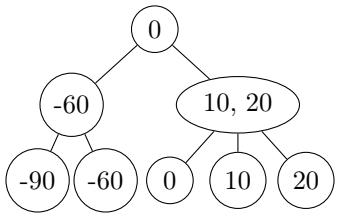
What must NOT be changed:

- Nothing in the application class should be changed, except as noted above.
- The headers of the `public` methods in the `Table` class must NOT be changed (except the constructor must lose its parameters, of course).
- The `Expression` class, the `Stack` class, the `Node` class, and the `Variable` class must not be changed.

Assignment Questions: Part B — 2-3 Trees

Insert -80, -5, 8, 4, and 7 (in that order) into the following 2-3 tree. Each successive insertion should start with the result of the previous insertion. For each insertion:

- State what L_{search} was (i.e., what is the value in the leaf at which the search ended?).
- Draw the 2-3 tree that results from the insertion.



Note: To draw a 2-3 tree, draw the leaves in a row first, leaving enough space for all the levels above the leaves. Then draw the parents of the leaves in a row, centering each parent above its children. Continue drawing the nodes on a level in a row, centering each parent above its children, level by level, up the tree to the root.

You must create a .pdf file containing your solution to this question. You can either create it electronically or you can hand-write your solution and scan it in. Either way, your answers must be clearly legible and well-organized so that the marker can easily mark it. This .pdf file must be named
`A5PartB<your last name><your student id>.pdf`.

Assignment Questions: Part C — Heaps

File `A5PartC.java` contains a nearly-complete application that reads in lists and sorts them using heap sort (which uses a heap). It uses the input file `inputFile5C.txt` (the file name is hard-coded into the application). Both `A5PartC.java` and `inputFile5C.txt` are available on UM Learn where you got this assignment.

You must complete the program by adding the bodies of a few methods in the `Heap` class:

- `leftChild( int i )`: This method returns the index of the left child of `heapArray[i]`. It does not do any checking if the left child exists, it simply computes the correct index for the left child.
- `rightChild( int i )`: This method returns the index of the right child of `heapArray[i]`. It does not do any checking if the right child exists, it simply computes the correct index for the right child.
- `deleteMax()`: This method removes and returns the highest priority item in the heap (in this heap, highest priority is the largest value). It should use at least one of the other `Heap` methods, so look at them to see which one(s) might be useful.

Finally, you must change the name of the file to `A5PartC<yourName><yourStudentID>.java` and you must change the name of the application class to match it.

Hand-in Instructions

Go to COMP2140 in UM Learn, then click "Dropbox" under "Assessments" at the top. You will find a dropbox folder called "Assignment 5". (If you cannot see the assignment dropbox, follow the directions under "Honesty Declaration" on Assignment 1.) Click the link and follow the instructions. Please note the following:

- **Submit ONE .zip file only. The .zip file must contain at most TWO .java files (your solutions to Part A and Part C) and ONE .pdf file (your solution to Part B), and nothing else.** The .zip file must be named `A5<your last name><your student id>.zip` (e.g., `A5Cameron1234567.zip`).
- Please do not submit anything else.
- You can submit as many times as you like, but only the most recent submission is kept.
- We only accept homework submissions via UM Learn. Please DO NOT try to email your homework to the instructor or TA or markers — it will not be accepted.
- We reserve the right to refuse to grade the homework or to deduct marks if you do not follow instructions.
- **Assignments become late immediately after the posted due date and time.** Late assignments will be accepted up to 49 hours after that time, at a penalty of 2% per hour or portion thereof. After 49 hours, an assignment is worth 0 marks and will no longer be accepted.